

Multicore Computer Architecture - Storage and Interconnects

Lecture 1

Instruction Execution Principles



NPTEL Online Course Multi-Core Computer Architecture and Interconnects Dr. John Jose IIT Guwahati

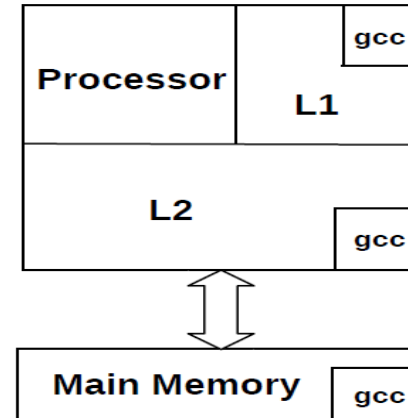
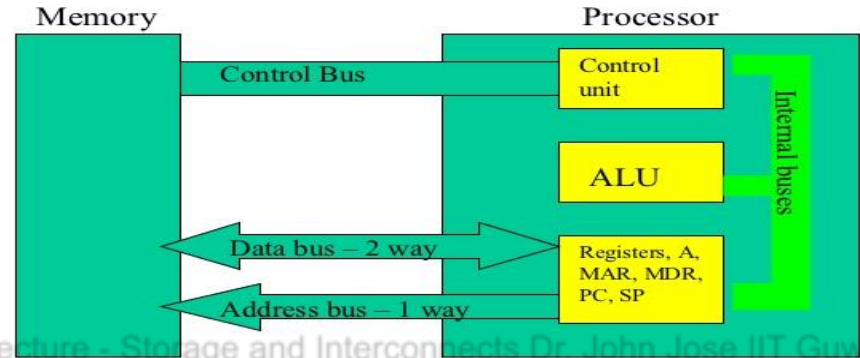
Dr. John Jose

Assistant Professor

**Department of Computer Science & Engineering
Indian Institute of Technology Guwahati, Assam.**

Processor Memory Interaction

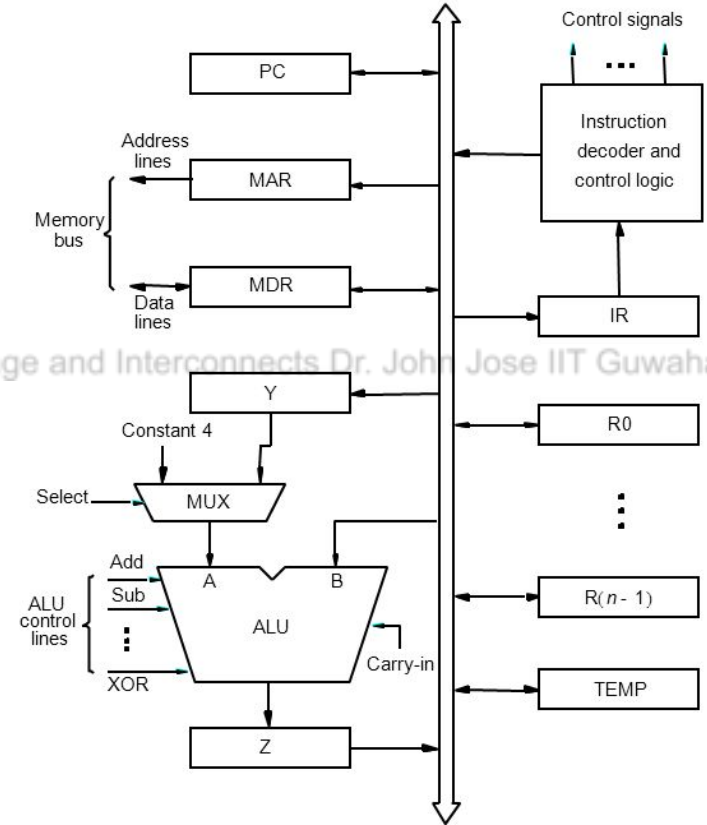
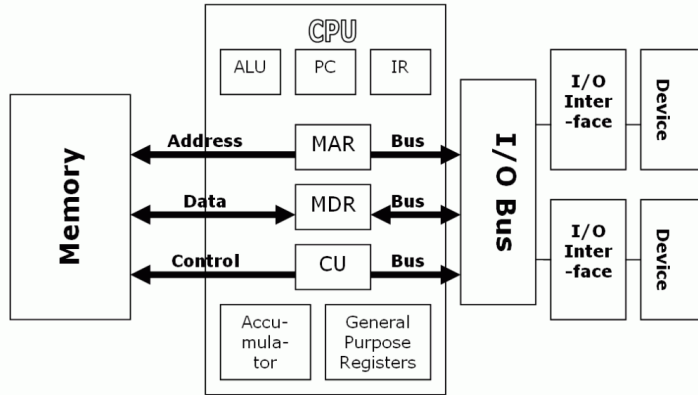
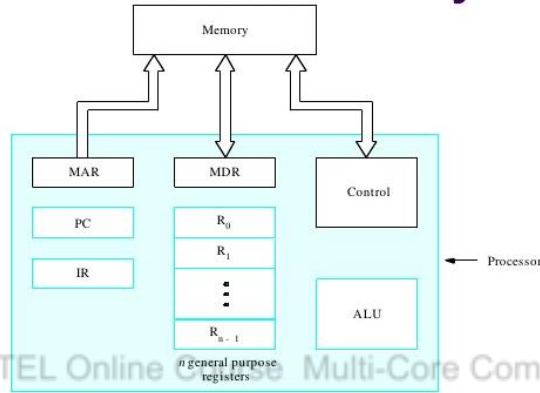
<i>Instruction Fetch</i>
<i>Instruction Decode</i>
<i>Operand Fetch</i>
<i>Execute</i>
<i>Result Store</i>
<i>Next Instruction</i>



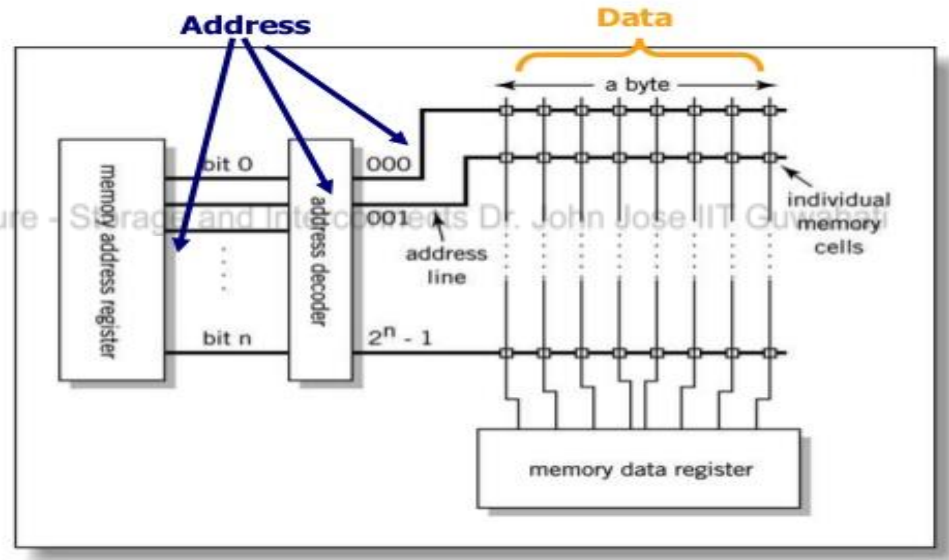
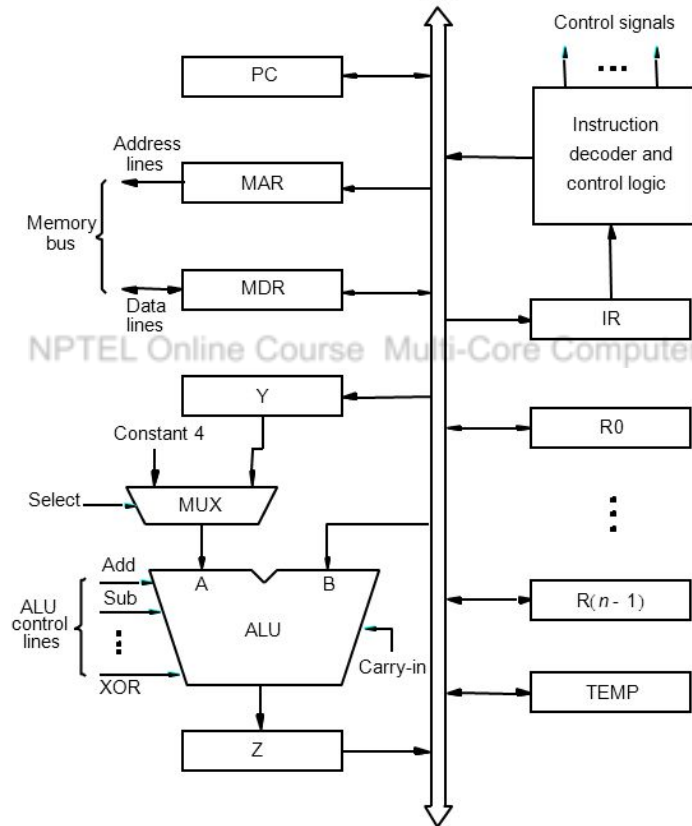
Execution Cycle

<i>Instruction Fetch</i>	Obtain instruction from program storage
<i>Instruction Decode</i>	Determine required actions and instruction size
<i>Operand Fetch</i>	Locate and obtain operand data
<i>Execute</i>	Compute result value or status
<i>Result Store</i>	Deposit results in storage for later use
<i>Next Instruction</i>	Determine successor instruction

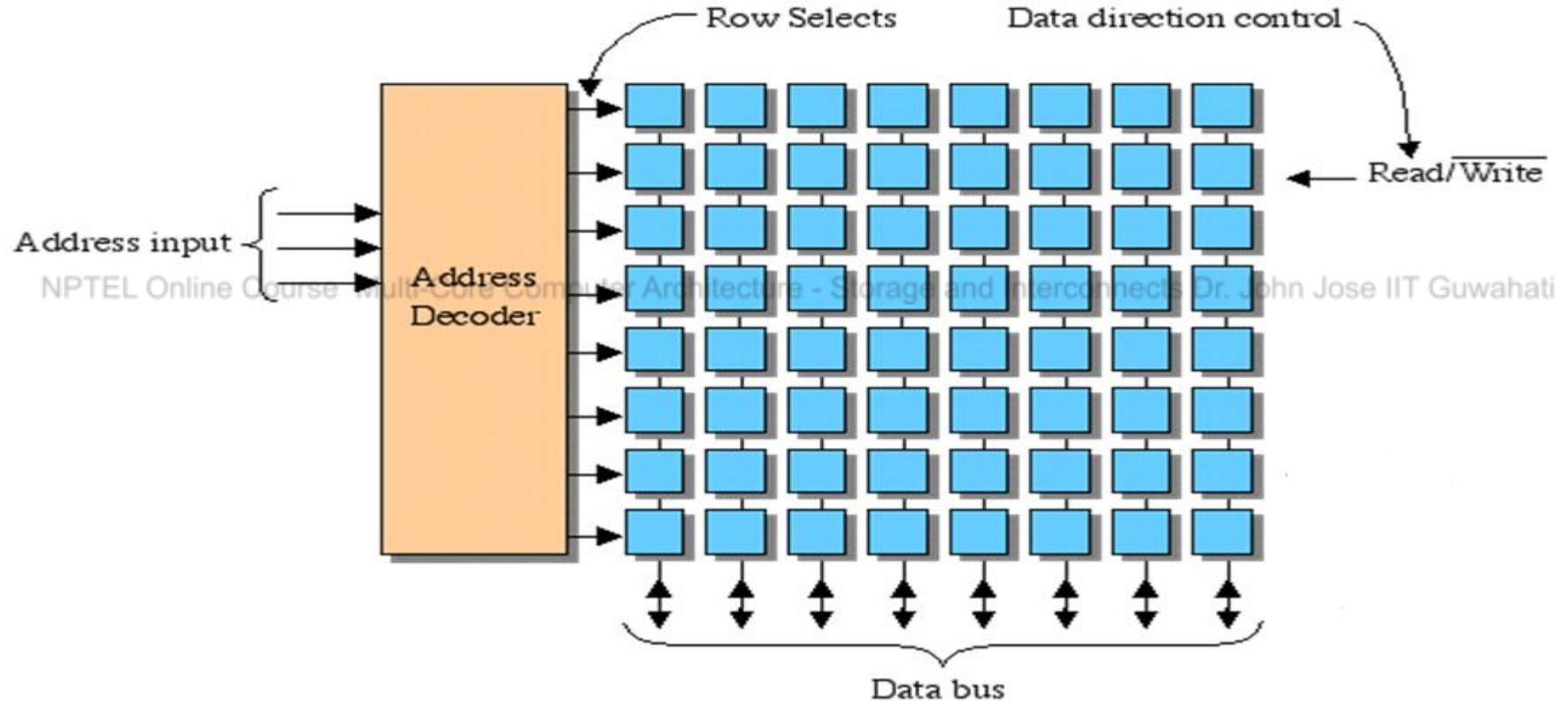
Processor Memory Interaction



Processor Memory Interaction

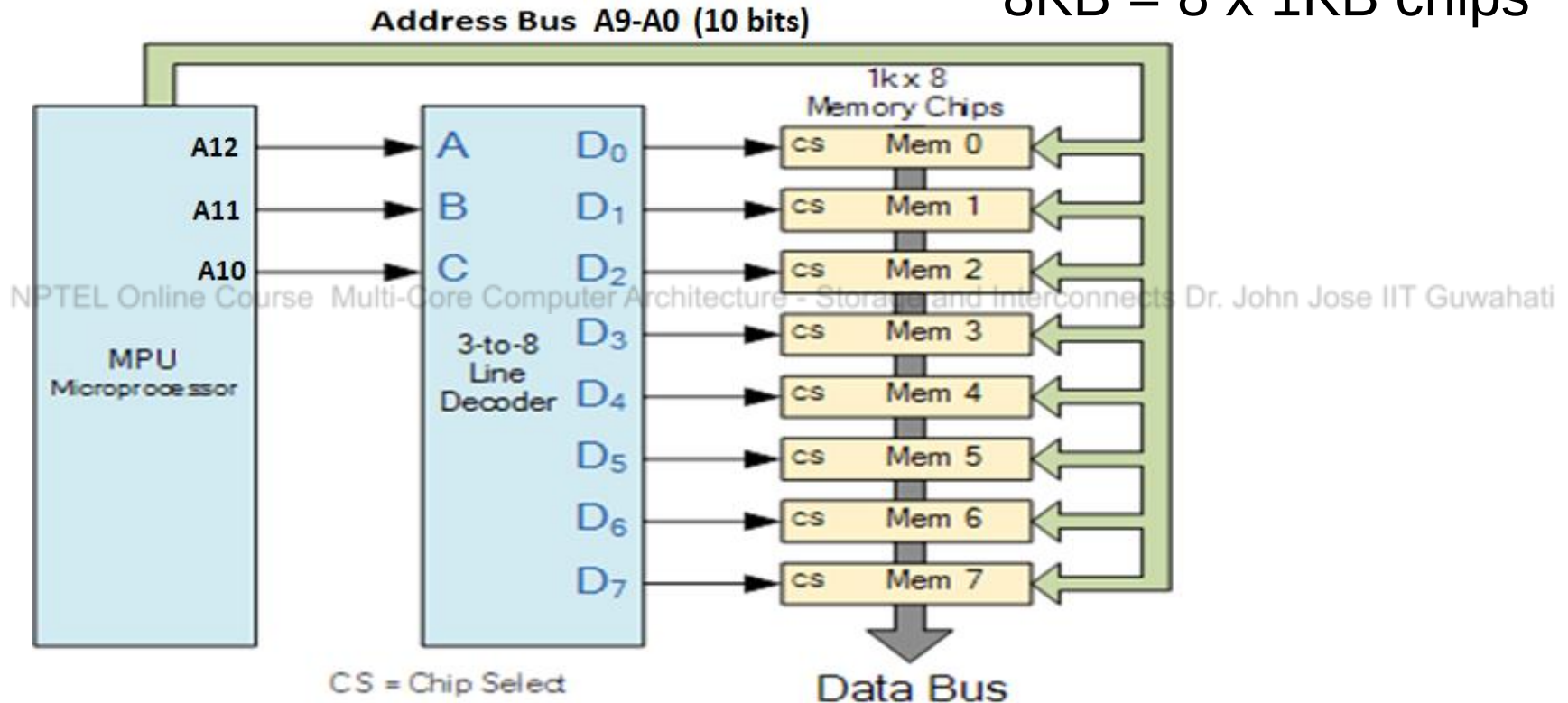


How memory works ?



How memory works ?

8KB = 8 x 1KB chips



Measuring Performance

❖ When can we say one computer / architecture design is better than others?

❖ Desktop PC : execution time of a program

❖ Server : transactions / unit time

NPTEL Online Course Multi-Core Computer Architecture - Storage and Interconnects Dr. John Jose IIT Guwahati

❖ When can we say X is n times faster than Y ?

❖ $\text{Execution time}_Y / \text{Execution time}_X = n$

❖ $\text{Throughput}_X / \text{Throughput}_Y = n$

Measuring Performance

❖ Typical performance metrics:

- ❖ Response time

- ❖ Throughput

- ❖ Execution time

- ❖ CPU computation time

❖ Benchmarks

- ❖ Synthetic benchmarks

- ❖ Benchmark suites (e.g. SPEC06, PARSEC, SPLASH)

Benchmark Suite

SPEC CPU2006 Programs

	Benchmark	Language	Descriptions
CINT2006 (Integer) 12 programs	400.perlbench	C	PERL Programming Language
	401.bzip2	C	Compression
	403.gcc	C	C Compiler
	429.mcf	C	Combinatorial Optimization
	445.gobmk	C	Artificial Intelligence: go
	456.hmmer	C	Search Gene Sequence
	458.sjeng	C	Artificial Intelligence: chess
	462.libquantum	C	Physics: Quantum Computing
	464.h264ref	C	Video Compression
	471.omnetpp	C++	Discrete Event Simulation
	473.astar	C++	Path-finding Algorithms
	483.Xalancbmk	C++	XML Processing
CFP2006 (Floating Point) 17 programs	410.bwaves	Fortran	Fluid Dynamics
	416.gamess	Fortran	Quantum Chemistry
	433.mile	C	Physics: Quantum Chromodynamics
	434.zeusmp	Fortran	Physics/CFD
	435.gromacs	C/Fortran	Biochemistry/Molecular Dynamics
	436.cactusADM	C/Fortran	Physics/General Relativity
	437.leslie3d	Fortran	Fluid Dynamics
	444.namd	C++	Biology/Molecular Dynamics
	447.dealII	C++	Finite Element Analysis
	450.soplex	C++	Linear Programming, Optimization
	453.povray	C++	Image Ray-tracing
	454.calculix	C/Fortran	Structural Mechanics
	459.GemsFDTD	Fortran	Computational Electromagnetics
	465.tonto	Fortran	Quantum Chemistry
	470.lbm	C	Fluid Dynamics
	481.wrf	C/Fortran	Weather Prediction
	482.sphinx3	C	Speech recognition

Target Programs application domain: Engineering and scientific computation

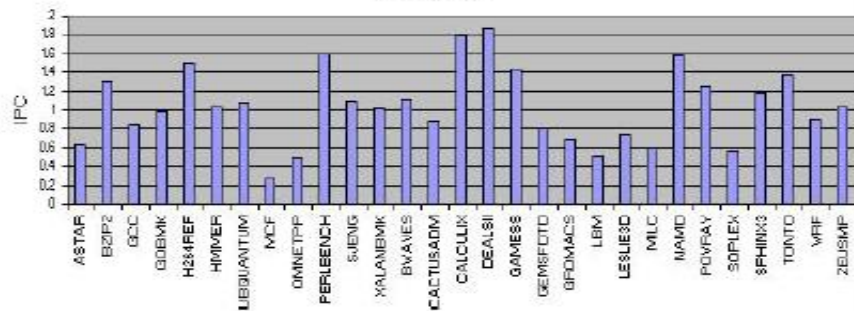
CMPE550 - Shaaban

Source: <http://www.spec.org/cpu2006/>

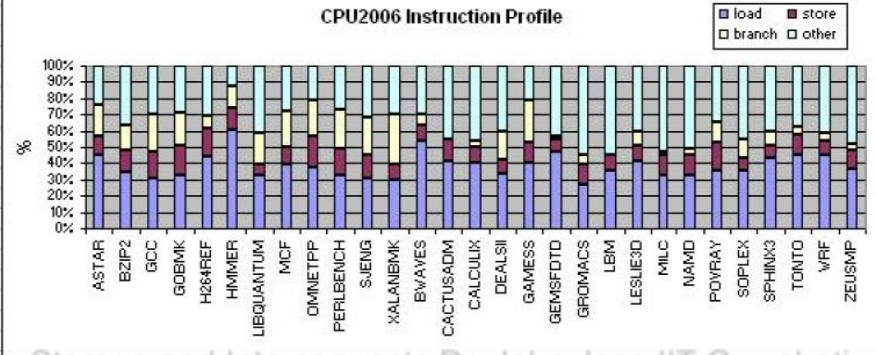
#39 Lec # 1 Spring 2015 1-26-2015

Benchmark based evaluation

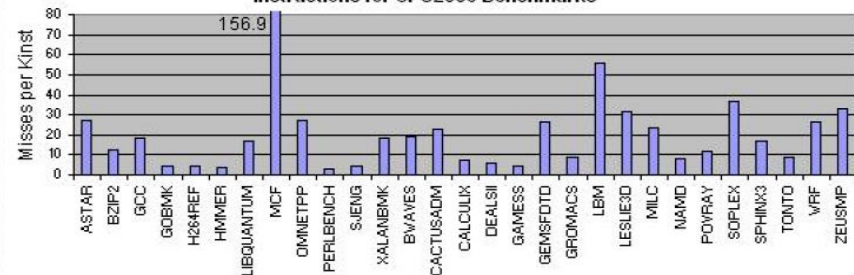
CPU2006 IPC



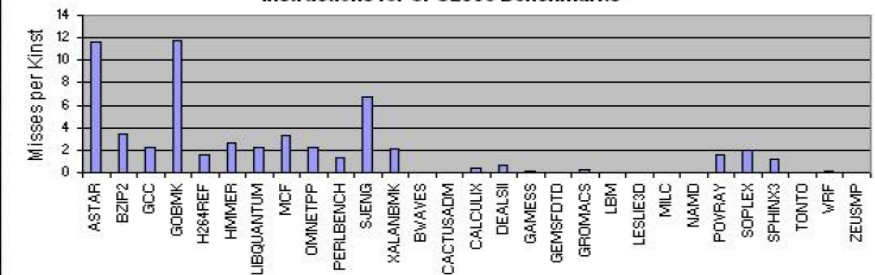
CPU2006 Instruction Profile



L1 Data Cache Misses per 1000 Instructions for CPU2006 Benchmarks



Branch Mispredictions per 1000 Instructions for CPU2006 Benchmarks



SPEC Ratio

$$\text{SPECRatio}_A = \frac{\text{Execution time}_{\text{reference}}}{\text{Execution time}_A}$$

Reference for SPEC 2006:

Sun Ultra Enterprise 2 workstation with
a 296-MHz UltraSPARC II processor

NPTEL Online Course Multi-Core Computer Architecture Storage and Interconnects Dr. John Jose IIT Guwahati

$$\frac{\text{SPECRatio}_A}{\text{SPECRatio}_B} = \frac{\frac{\text{Execution time}_{\text{reference}}}{\text{Execution time}_A}}{\frac{\text{Execution time}_{\text{reference}}}{\text{Execution time}_B}} = \frac{\text{Execution time}_B}{\text{Execution time}_A} = \frac{\text{Performance}_A}{\text{Performance}_B}$$

$$\text{GeometricMean}(a_1, a_2, a_3, \dots, a_N) = \sqrt[N]{\prod_i a_i}$$

Amdahl's Law

- ❖ Amdahl's Law defines the speedup that can be gained by improving some portion of a computer.
- ❖ **The performance improvement to be gained from using some faster mode of execution is limited by the fraction of the time the faster mode can be used.**

NPTEL Online Course Multi-Core Computer Architecture - Storage and Interconnects Dr. John Jose IIT Guwahati

$$\text{Execution time}_{\text{new}} = \text{Execution time}_{\text{old}} \times \left((1 - \text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}} \right)$$

$$\text{Speedup}_{\text{overall}} = \frac{\text{Execution time}_{\text{old}}}{\text{Execution time}_{\text{new}}} = \frac{1}{(1 - \text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}}$$

Amdahl's Law- Illustration

Example: Suppose that we want to enhance the floating point operations of a processor by introducing a new advanced FPU unit. Let the new FPU is 10 times faster on floating point computations than the original processor. Assuming a program has 40% floating point operations, what is the overall speedup gained by incorporating the enhancement?

Solution:

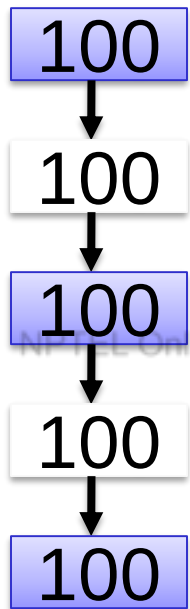
Fraction enhanced = 0.4

Speedup enhanced = 10

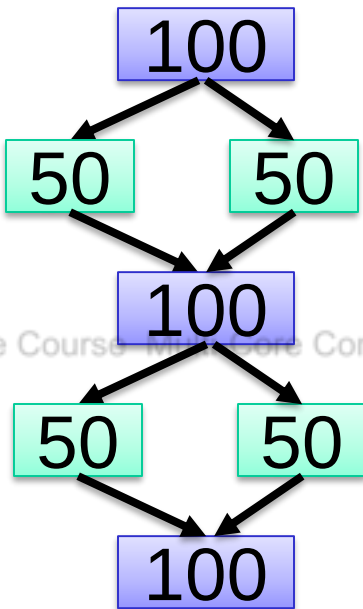
$$\text{Speedup}_{\text{overall}} = \frac{1}{(1 - \text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}}$$

$$\text{Speedup}_{\text{overall}} = \frac{1}{0.6 + \frac{0.4}{10}} = \frac{1}{0.64} \approx 1.56$$

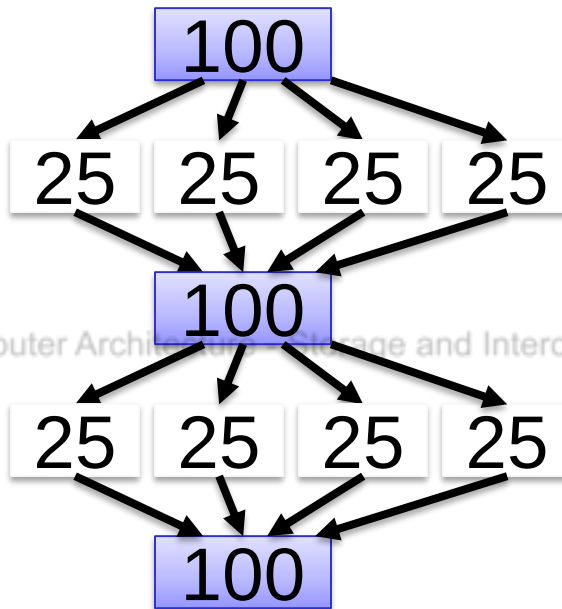
Amdahl's Law for multi-cores



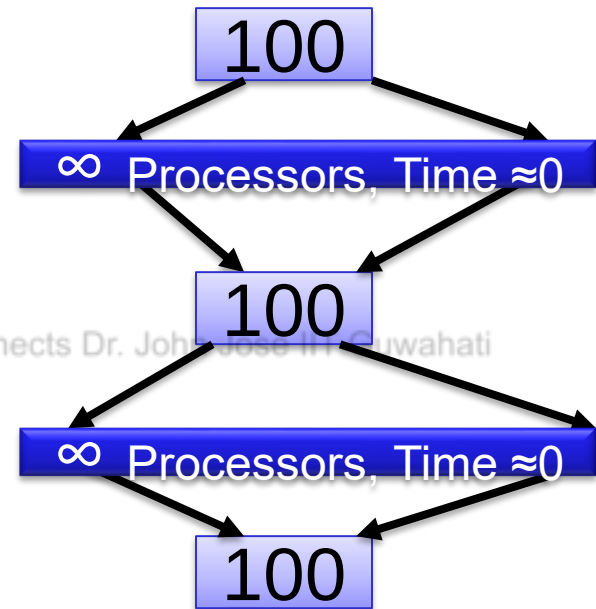
Work 500,
Time 500
Sp=1X



Work 500,
Time 400
Sp=1.25X



Work 500,
Time 350
Sp=1.4X



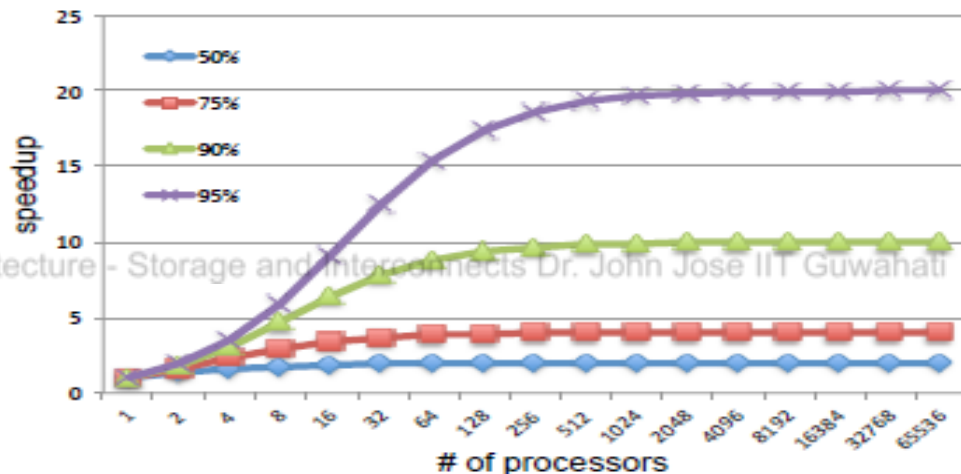
Work 500,
Time 300
Sp=1.7X

How much Speed up you can achieve ?

Amdahl's Law:

NPTEL Online Course Multi-Core Computer Architecture - Storage and Interconnects Dr. John Jose IIT Guwahati

$$Speedup = \frac{1}{(1 - \alpha) + \frac{\alpha}{n}}$$



Instruction Set Architecture

- ❖ **Instruction vs Program vs Software**

- ❖ **Opcode, Operand**

- ❖ **Classification of ISA**

NPTEL Online Course Multi-Core Computer Architecture - Storage and Interconnects Dr. John Jose IIT Guwahati

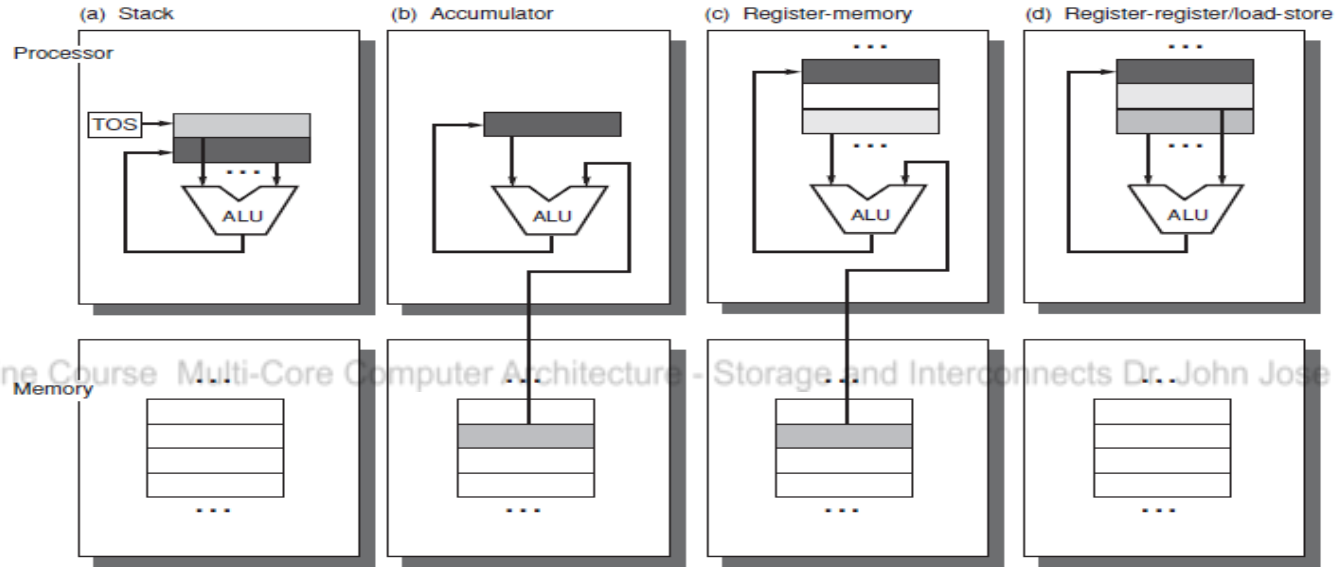
 - ❖ **Stack architecture**

 - ❖ **Accumulator architecture**

 - ❖ **Register-Memory architecture**

 - ❖ **Register-Register/Load Store architecture**

Instruction Set Architecture

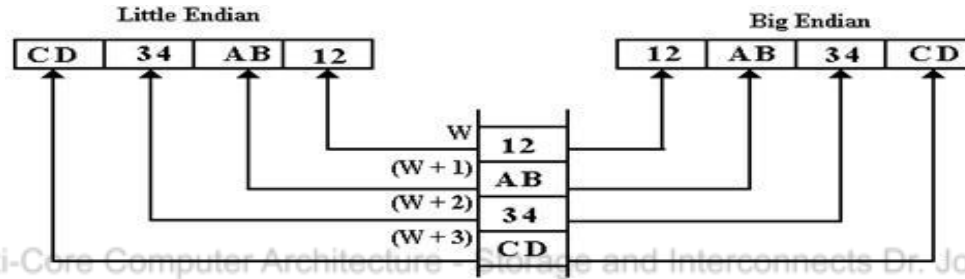


Stack	Accumulator	Register (register-memory)	Register (load-store)
Push A	Load A	Load R1,A	Load R1,A
Push B	Add B	Add R3,R1,B	Load R2,B
Add	Store C	Store R3,C	Add R3,R1,R2
Pop C			Store R3,C

The code sequence for $C = A + B$ for four classes of instruction sets.

Byte Ordering

❖ Big Endian vs Little Endian

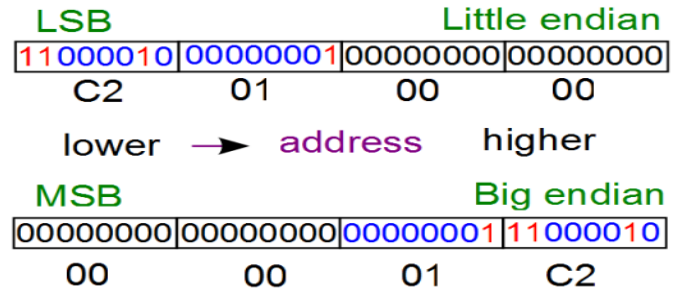


NPTEL Online Course Multi-Core Computer Architecture Storage and Interconnects Dr. John Jose IIT Guwahati

Big Endian vs. Little Endian



$$\text{Int } i = 450 = 2^8 + 2^7 + 2^6 + 2 = \text{x000001C2}$$



Byte / Word Alignment

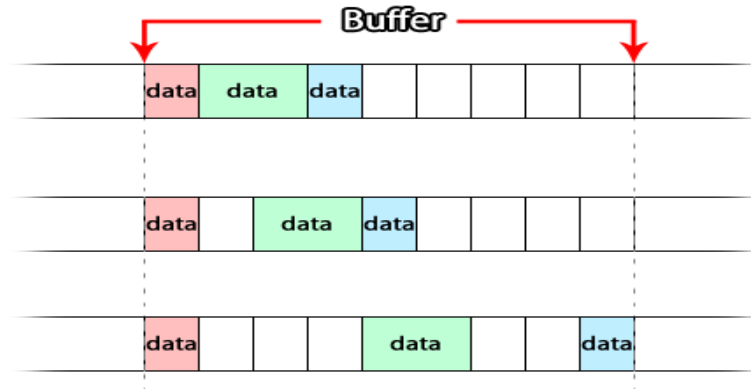
Short			Short			Short			...		
X	Y	Z	Nx	Ny	Nz						
0	1	2	3	4	5	6	7	8	9	10	11

Misaligned (slow)



Add padding bytes so that attributes start at 4-byte boundaries

Short			Short			Short			Short			Short			Short		
X			Y			Z			Nx			Ny			Nz		
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15		



Principles of Computer Design

- ❖ All processors are driven by clock.
- ❖ Expressed as clock rate in GHz or clock period in ns

$\text{CPU time} = \text{CPU clock cycles for a program} \times \text{Clock cycle time}$

$$\text{CPI} = \frac{\text{CPU clock cycles for a program}}{\text{Instruction count}}$$

NPTEL Online Course Multi-Core Computer Architecture - Storage and Interconnects Dr. John Jose IIT Guwahati

$\text{CPU time} = \text{Instruction count} \times \text{Cycles per instruction} \times \text{Clock cycle time}$

$$\frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock cycle}} = \frac{\text{Seconds}}{\text{Program}} = \text{CPU time}$$

- ❖ Clock cycle time - hardware technology
- ❖ Cycles/Instruction - organization and ISA
- ❖ Instruction Count -ISA and compiler technology

Principles of Computer Design

- ❖ Different instruction types having different CPIs

$$\text{CPU clock cycles} = \sum_{i=1}^n \text{IC}_i \times \text{CPI}_i$$

$$\text{CPU time} = \left(\sum_{i=1}^n \text{IC}_i \times \text{CPI}_i \right) \times \text{Clock cycle time}$$

$$\text{CPI} = \frac{\sum_{i=1}^n \text{IC}_i \times \text{CPI}_i}{\text{Instruction count}}$$

Design Example

A common transformation required in graphics processors is square root. Implementations of floating-point (FP) square root vary significantly in performance, especially among processors designed for graphics. Suppose FP square root (FPSQR) is responsible for 20% of the execution time of a critical graphics benchmark.

One proposal is to enhance the FPSQR hardware and speed up this operation by a factor of 10. The other alternative is just to try to make all FP instructions in the graphics processor run faster by a factor of 1.6; FP instructions are responsible for half of the execution time for the application. Compare these two design alternatives using Amdahl's Law.

Reading Exercises

❖ **Computer Architecture-A Quantitative Approach** (5th edition),

John L. Hennessy, David A. Patterson, Morgan Kaufman.

❖ Chapter 1.8 – Measuring, Reporting and summarizing Performance

NPTEL Online Course Multi-Core Computer Architecture - Storage and Interconnects Dr. John Jose IIT Guwahati

❖ Chapter 1.9 – Quantitative Principles of Computer Design

**THANK
YOU!**

**FOR
YOUR
PRESENCE**

NPTEL Online Course Multi-Core Computer Architecture Storage and Interconnects Dr. John Jose IIT Guwahati

johnjose@iitg.ac.in
<http://www.iitg.ac.in/johnjose/>